A Prototype Node for Wireless Vision Sensor Network Applications Development

M. Bakkali (*), R. Carmona-Galán(**), A. Rodriguez-Vazquez(**)

(*) Anafocus-Universidad de Sevilla
Av. Isaac Newton nº4,Pabellón de Italia, 7ª Planta
Parque Tecnológico Isla de la Cartuja
41092 Sevilla, Spain
Email: mbakkali@anafocus.com

(**) Instituto de Microelectrónica de Sevilla–CNM
CSIC-Universidad de Sevilla, C/Américo Vespucio s/n
Parque Tecnológico Isla de la Cartuja
41092 Sevilla, Spain
Email: {rcarmona,angel}@imse-cnm.csic.es

***Abstract*—This paper presents a prototype vision-enabled sensor node based on a commercial vision system of reduced size and power consumption. The wireless infrastructure for the deployment of a distributed smart camera network based on these nodes is provided by commercial motes. The smart camera, based on a low-power bio-inspired processing scheme, enables in-node image processing and vision tools. This permits to elaborate a lighter representation of the scene, keeping the relevant information in terms of detected elements, features and events, alleviating the data transmission through the network. Therefore by passing only the relevant information to the neighboring sensor nodes, distributed and collaborative vision is possible with the limited data rates available in commercial wireless sensor networks. Communication between the different components of the system is supported by the available UARTs and GPIOs. Several examples of in-node image processing and feature detection has been tested in the prototype, and information at different abstraction levels has been broadcasted to the network.**

Keywords: Wireless Sensor Network (WSN), Wireless Vision Sensor Network (WVSN), Smart Camera.

I. INTRODUCTION

Distributed smart camera applications [1] seem to be an appropriate scenario in which highly integrated vision systems, oriented towards autonomous low-power operation, can display their outstanding potential. The functionality of a vision-enabled sensor node goes beyond the mere acquisition and transmission of images. In fact, the required bandwidth for simultaneously handling the image flow generated by a network of distributed cameras can be prohibitive [2]. Therefore, the smart camera surveys its visual field and analyzes the events that take place under its sight. This analysis can be local or in cooperation with other smart cameras. Objects are recognized and singular events are detected and time stamped. The final outcome of the operation of the smart camera can be a well elaborated response, consisting in the activation of actuators or the transmission of an alarm code [3].

Several camera-mote systems have been reported in the last years (Table I). In most of the cases, a compact CMOS image sensor module is attached to a microprocessor board with a wireless communication module, like in *MeshEye* [4] or the wireless-enabled *CMUcam3* [5]. These represent the most compact designs, although in-node image processing is restricted by the computing power that can be provided by their processing architecture. Other reported systems make use of available wireless motes to support the sensor network, like *Cyclops* [6] and *CITRIC* [7]. Again, the processing scheme consists in image capture, A/D conversion and then processing with a general purpose microprocessor. In order to implement the computational demand found in the early stages of vision, when a vast amount of raw image data has to be processed, this architecture can be quite inefficient. On one hand, the energy spent in the symbolic representation of data and their processing in the digital domain can be a useless waste when only low to moderate accuracy is needed. On the other hand, repeated access to and from memory also represents a quite significant consumption. *WiCa* [8] is an alternative approach in which the processing scheme has been adapted to the nature of the visual stimulus. It is based on a processor with a dedicated image processing architecture, the *Xetal-II* [9], a massively parallel SIMD with 320 processing elements.

TABLE I. CAMERA-MOTE SYSTEMS

Ref.	Year	Camera	Processor	Comm.
[6]	2004	320×240px ADCM1700	ATmega128 @7.3 MHz	ZigBee IEEE 802.15.4
[8]	2006	640×480px	Xetal-II: SIMD 320 PEs @80 MHz	ZigBee IEEE 802.15.4
[4]	2007	2×640×480px ADCM2700	AT91SAM7S (ARM7) @55 MHz	ZigBee IEEE 802.15.4
[5]	2007	352×288px OV6620	NXP-LPC2106 (ARM7) @60 MHz	ZigBee IEEE 802.15.4
[7]	2008	1280×1024px OV9655	Intel PXA270 (ARM9) @520 MHz	ZigBee IEEE 802.15.4

Our approach to in-node image processing is bio-inspired, consisting in the sense that the system is adapted to the nature of the stimuli at two levels. First, we are leaving the most tedious processing tasks —those with a regular computational flow that is evenly applied to each pixel of the image—, to an array of moderately accurate processing elements that are concurrent with the photosensors. This is precisely the way in which the biological retina is organized [10]. They operate in parallel and contain local memories in order to avoid large consumption overheads due to memory access. Second, operations in the image plane, also referred as early vision tasks, are realized by analog circuits, rendering enough accuracy for low power consumption. This type of visual microprocessor is at the core of the *Eye-RIS* [11], which is the smart camera —fabricated by Anafocus [12], at our vision-enabled wireless sensor node. The last generation sensor/pre-processor used at the *Eye-RIS*, named *Q-Eye*, exhibits a computational power of 250GOPS with a power consumption of 4mW/GOPS. This figure is in the order of the peak performance reported for *Xetal-II* [9] that is 5.6mW/GOPS.

With regard to the network infrastructure, we are using a commercial solution —fabricated by Crossbow [13], for the development of wireless sensor network applications. These motes, autonomous systems with environmental scalar sensors and processing and communication capabilities, are linked to the *Eye-RIS* constituting a prototype node of a wireless distributed vision network. In order to establish the necessary interconnection between the vision processing subsystem and the wireless communication link, we have explored the different ports available at each device. As an example, we have implemented a motion detection algorithm at the sensor node that can trigger an alarm, broadcast the coordinates of the detected events in the scene, or analyze the characteristics of the detected changes and send this information over the network.

II. EYE-RIS SMART CAMERA DESCRIPTION

The *Eye-RIS* is a programmable autonomous vision system that employs a novel architecture in which image-processing is accomplished following a hierarchical approach with two levels (Fig. 1). Right after image acquisition, *early image processing* takes place. The basic tasks at this level are meant to extract the useful information from the input image flow. Therefore, the output is a reduced set of data comprising image features. These tasks are realized by the *Q-Eye* smart image sensor [11] which consists of an array of 176×144 interconnected sensing-processing cells. The second processing level, where the data amount is smaller although of a higher abstraction level, is implemented by a RISC processor. In particular, by *NIOS-II*, a 32-bit embedded processor designed specifically for the Altera family of FPGAs. The tasks realized meant to output complex decisions and to support action-taking. These tasks may involve complex algorithms with long and irregular computational flows and may require greater accuracy than early processing.

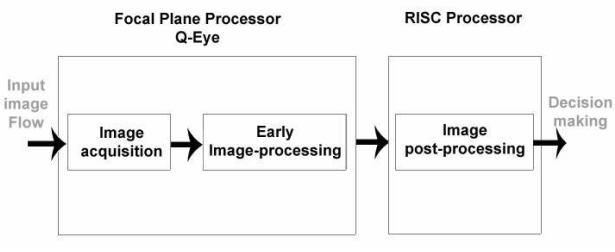


Figure 1. *Eye-RIS* vision systems functional diagram

Due to this scheme, the *Eye-RIS* achieves a high processing speed with moderate power consumption. It can be employed for applications in which acquisition-processing rates can reach beyond 500fps. This is difficult to accomplish with a conventional processing scheme, e. g. the 50fps achieved by the *CMUcam3* [5].

III. WIRELESS COMMUNICATIONS SUPPORT

In our system, wireless communications are realized through an *IRIS XM2110* mote [13]. This mote features 250Kbps data rate, and outdoor line-of-sight ranges as far as 500m between nodes without amplification. The selection of a commercial platform is mainly based on the availability, together with a matured and tested hardware, of a wide set of software tools. Namely: an operating system and a programming language and compiler [14] for the mote microprocessor (*ATmega1281*), a software environment for mote programming and network monitoring and application development (*MoteWorks*). Also, this module is smart enough to avoid the need to handle the communication protocol at the camera, leaving more resources to on-site image processing.

A. Camera- mote GPIO link

The most direct way to communicate the smart camera (*Eye-RIS*) with the mote is using the available general purpose I/O pins (GPIO). Inside the vision system, the *NIOS-II* can address these GPIOs and send signals through them at points in time determined by the running algorithm. At the mote side, the corresponding GPIOs are being continuously scanned in order to detect any change. The occurrence of the appropriate event can be employed to trigger actions into the mote like initiating a message broadcast.

B. Camera-mote communication through UART

Both subsystems can communicate bytes, instead of signals. It can be done by using the available UARTs. In this occasion, several adjustments are required. First, the *Eye-RIS*' UART uses the RTS/CTS method to control the data flow. This is not implemented in the mote, thus, data interchange between them is initially disabled. This can be overcome by forcing the RTS pin in the *Eye-RIS* to a high-state (5V) at the expense of the handshaking capabilities (Fig. 2). Another difficulty found is in the voltage levels representing the different logic states. An adapter chip is employed for this (*MAX3222*). Also the baud rate has to be set to a compatible level, 115200 baud.

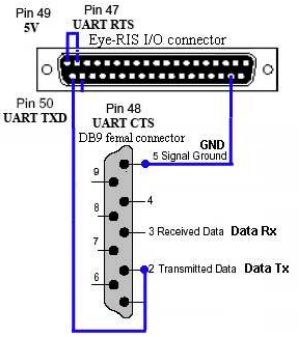


Figure 2. Connection between Eye-RIS UART and RS232 connector

IV. APPLICATION EXAMPLES

A. *Motion-triggered broadcast of an alarm signal*

As an elementary demonstrator of the system capabilities for surveillance and monitoring of, for instance, public spaces, we have tested several routines to detect changes in the scene and send related information over the network. The basic algorithm consists in a simple image change detection running on the *Eye-RIS* vision system, and a signal being transmitted to one of the GPIOs. This pin is connected to one of the mote GPIOs, thus triggering the broadcast of an alarm over the air. We have employed other two motes in this elementary network (Fig. 3). One of them receives the alarm signal and acknowledges the reception, thus verifying the wireless communication link. The other mote, that is part of the base station, runs a network-traffic sniffing tool that allows broadcast monitoring.

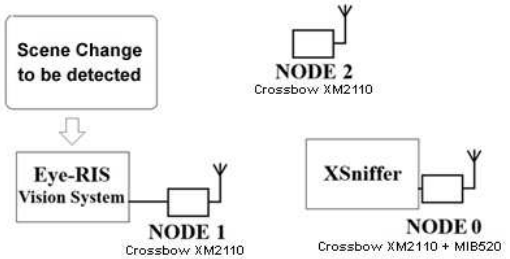


Figure 3. Conceptual diagram for the sample application setup

The algorithm designed to illustrate the capabilities of the *Eye-RIS*-mote system as a prototype the development of wireless vision sensor network applications is quite basic. Fig. 4 depicts the algorithm running on the *Eye-RIS*. It consists in a simple difference method [15]. Consider that the sensor is acquiring an image flow represented by function:

$$I = I(x, y, t): \mathbb{R}^3 \rightarrow \mathbb{R} \qquad (1)$$

where x and y are the coordinates of the pixel in the image plane and t is the tie instant in which the frame is captured. Of course, these variables are discretized in space, because of the measurable size of the image points, and in time, because image frames are captured at intervals T_S, which is the sampling period. Let I_i be the frame taken at $t_i = t_0 + i\,T_S$ and I_j the one taken at $t_j = t_0 + j\,T_S$. These frames are separated in the flow by time interval Δt given by:

$$\Delta t = (j - i)T_S \qquad (2)$$

Notice that this waiting period is introduced to encode the actual speed of the changes that we would like to capture. Then, the absolute difference of the two frames is computed:

$$I_{diff}(x, y) = \left| I(x, y, t_i + \Delta t) - I(x, y, t_i) \right| \qquad (3)$$

and finally a threshold function is applied to eliminate those pixels where the change is not significant and probably due to spurious differences:

$$I_{out}(x, y) = \begin{cases} 1 & I_{diff}(x, y) > T \\ 0 & I_{diff}(x, y) \le T \end{cases} \qquad (4)$$

All these steps have been performed at the image plane by making use of the operators available at the focal-plane in the *Q-Eye*. In our first example, that is intended to trigger an alarm is any relevant changes occur in the scene, the operation at the focal-plan concludes with an activity scan, also provided as a function of the *Eye-RIS* image processing library as there is some dedicated hardware included at the pixel-level to speed up this scan. If the result of the activity scan is positive then a signal is set to one of the GPIOs. The mote at node 1 (Fig. 3), that was repeatedly sampling this GPIO in the meantime (Fig. 5), detects the signal and transmit the alarm. The blink of the red led in node 2 (Fig. 3) indicates that the scene changes have been remotely sensed.

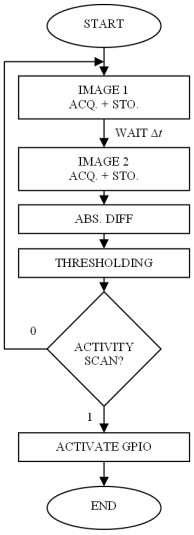
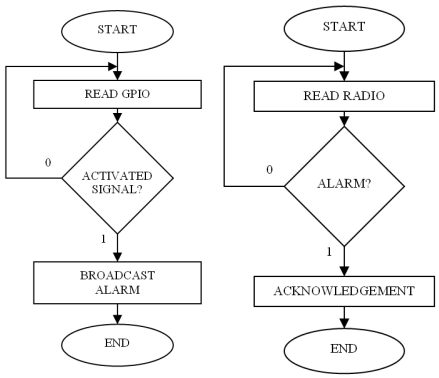


Figure 4. Flowchart of the vision system algorithm

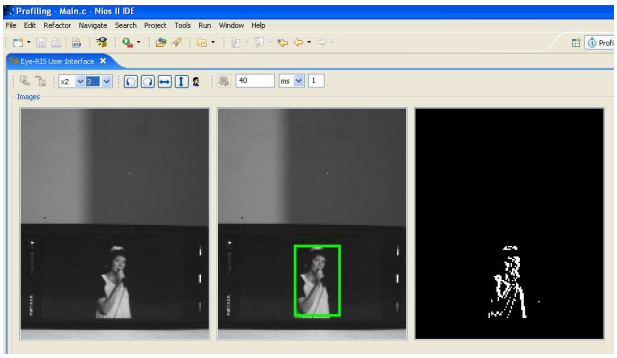


(a) (b)

Figure 5. Algorithms running in the motes: (a) node 1 and (b) node

B. *Broadcast of the coordinates of the ROI*

The *Eye-RIS* can deliver not only simple signals, but any information extracted from the image and processed and formatted by the *NIOS-II*. For instance, we can define a region of interest (ROI), by tracing a bounding box around the area in which a significant of motion was detected. This is easily achieved with advanced scan functions like the activity scan previously mentioned. Now, instead of delivering a simple alarm, packets of up to 29 bytes can be built at the mote and then transmitted over the network.



(a) (b) (c)

Figure 6. (a) Original image, (b) highlighted ROI and (c) thresholded image difference at the *Eye-RIS* application development tool

C. *Broadcast of the in-situ image analysis results*

In order to illustrate the capabilities provided to the sensor nodes, we have tested the in-situ analysis of the image flow. Therefore, by storing the coordinates of the previously detected ROIs, the path of the moving objects can be tracked and information about the nature of the motion can be elaborated and delivered. In this occasion, we are sending a code indicating the sense of the major motion component (Fig. 7).

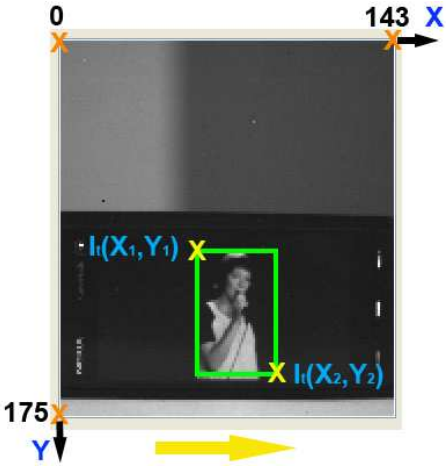


Figure 7. Motion sense detection at the *Eye-RIS*

V. CONCLUSIONS

A prototype system comprising a low-power smart camera and a wireless mote has been built. The bio-inspired approach to image processing renders competitive figures in terms of power efficiency given the high computational demand of vision algorithms. Technical difficulties related with the interconnection of heterogeneous modules have been solved. With the help of some application examples, the possibilities of the system to be employed as a smart node of a vision-enabled wireless sensor network have been demonstrated.

ACKNOWLEDGMENTS

This work is partially funded by JA/CICE through grant 2006-TIC-2352 and the PIMA program, and also by MICINN through grant TEC2009-11812, and co-funded by FEDER.

REFERENCES

[1] *Proceeding of the IEEE*, Vol. 96, no. 10, October 2008.

[2] G. Pekhteryev, Z. Sahinoglu, P. Orlik, G. Bhatti, "Image Transmission Over IEEE 802.15.4 and ZigBee Networks". *Proc. IEEE Int. Symp. Circuits and Systems*, Vol. 4, pp. 3539-3542, May 2005.

[3] M. Bramberger, A. Doblander, A. Maier et al., "Distributed Embedded Smart Cameras for Surveillance Applications". *Computer*, Vol. 39, No. 2, pp. 68-75, February 2006.

[4] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan, "MeshEye: A Hybrid-Resolution Smart Camera Mote for Applications in Distributed Intelligent Surveillance". *Int. Symp. Information Processing in Sensor Networks*, pp. 360-369, April 2007.

[5] A. Rowe, D. Goel, R. Rajkumar, "FireFly Mosaic: A Vision-Enabled Wireless Sensor Networking System". *Proc. 28th IEEE Int. Real-Time Systems Symposium*, pp. 459-468, Dec. 2007.

[6] M. Rahimi, R. Baer, O.I. Iroezi, J.C. Garcia, J. Warrior, D. Estrin, M. Srivastava, "Cyclops: In Situ Image Sensing and Interpretation in Wireless Sensor Networks". *Proc. 3rd ACM Int. Conf. Embedded Networked Sensor Systems*, pp. 192-204, Nov. 2005

[7] P. Chen, P. Ahammad, C. Boyer, et al. "CITRIC: A Low-Banwidth Wireless Camera Network Platform". *Proc. ACM/IEEE Int. Conf. Distributed Smart Cameras*, pp. 1-10, Sept. 2008

[8] R. Kleihorst, A. Abbo, B. Schueler, and A. Danilin, "Camera Mote with a High-Performance Parallel Processor for Real-Time Frame-Based Video Processing". *Proc. ACM/IEEE Int. Conf. Distributed Smart Cameras*, pp. 69-74, Sept. 2007.

[9] A. A. Abbo, R. P. Kleihorst, et al. "Xetal-II: A 107 GOPS, 600 mW Massively Parallel Processor for Video Scene Analysis". *IEEE J. Solid-State Circuits,* Vol. 43, No. 1, pp. 192-201, Jan. 2008.

[10] R. H. Masland, "The Fundamental Plan of the Retina", *Nature Neuroscience*, Vol. 4, pp. 877-886, 2001.

[11] A. Rodríguez-Vázquez et al., "The Eye-RIS CMOS Vision System", in H. Casier et al. (eds.), *Analog Circuit Design*. Springer, 2008.

[12] http://www.anafocus.com/

[13] http:// www.xbow.com/

[14] Philip Levis, *TinyOS/nesC Programming Reference Manual*, Crossbow Inc. January 2006.

[15] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image Change Detection Algorithms: A Systematic Survey". *IEEE Trans. Image Processing,* Vol. 14, No. 3, pp. 294-307, March 2005.